Geospatial RDF Data

Indexing and Query Answering for Geospatial in bigdata®

Index

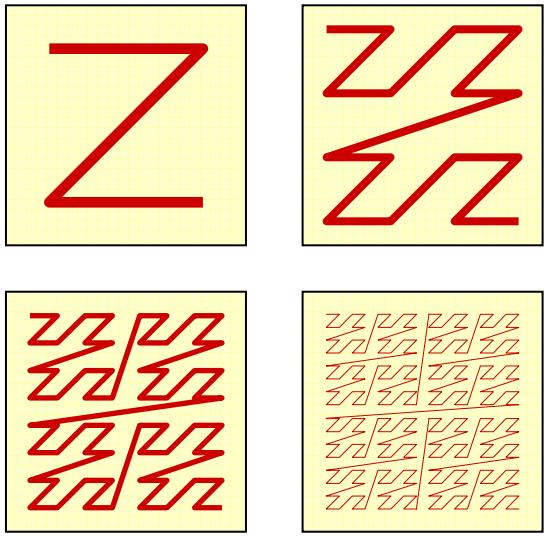
1. Why Geospatial With RDF?
2. Background: Z-Ordering
3. Representing Geospatial Data in RDF
4. Geospatial Indexing
5. Basic Geospatial Query
6. Geospatial Joins in SPARQL

Why Geospatial With RDF?

- RDF allows for near real-time data integration across data providers
  - Schema flexible, dynamic data model, indexed for joins
- Geospatial integration in RDF will enable mashups across geospatially-aware data sets
  - Social networking applications
  - Cross-organizational data platforms
  - Logistics
  - Command and control

What is Z-Ordering?

- “Space-filling curve”
- Maps multidimensional data to one dimension with good locality
  - Works for N-dimensions
- One-dimensional Z-ordered data can then be used in B-Tree indices
- Z-value calculated by interleaving binary representations of coordinate values:



Four iterations of a Z-curve

(x,y)	x	x(bin)	y	y(bin)	Z(bin)	Z
(1,3)	1	001	3	011	001011	11
(3,4)	3	011	4	100	100101	37
(5,2)	5	101	2	010	011001	25

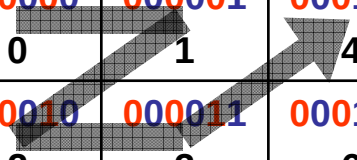
What is Z-Ordering?

	000 0	001 1	010 2	011 3	100 4	101 5	110 6	111 7
000 0	000000 0	000001 1	000100 4	000101 5	010000 16	010001 17	010100 20	010101 21
001 1	000010 2	000011 3	000110 6	000111 7	010010 18	010011 19	010110 22	010111 23
010 2	001000 8	001001 9	001100 12	001101 13	011000 24	011001 25	011100 28	011101 29
011 3	001010 10	001011 11	001110 14	001111 15	011010 26	011011 27	011110 30	011111 31
100 4	100000 32	100001 33	100100 36	100101 37	110000 48	110001 49	110100 52	110101 53
101 5	100010 34	100011 35	100110 38	100111 39	110010 50	110011 51	110110 54	110111 55
110 6	101000 40	101001 41	101100 44	101101 45	111000 56	111001 57	111100 60	111101 61
111 7	101010 42	101011 43	101110 46	101111 47	111010 58	111011 59	111110 62	111111 63

X

Y

What is Z-Ordering?

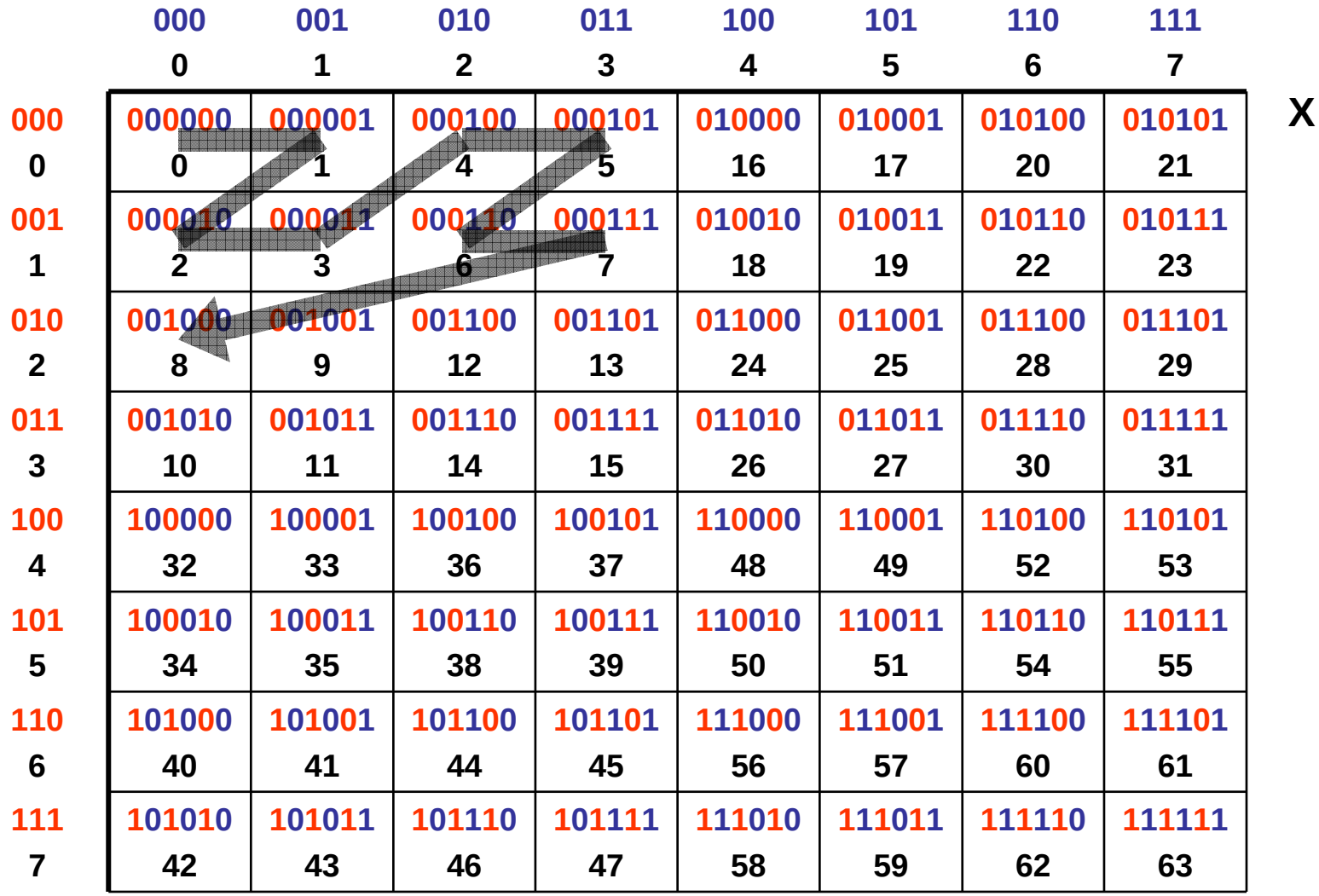


	000 0	001 1	010 2	011 3	100 4	101 5	110 6	111 7
000 0	000000 0	000001 1	000100 4	000101 5	010000 16	010001 17	010100 20	010101 21
001 1	000010 2	000011 3	000110 6	000111 7	010010 18	010011 19	010110 22	010111 23
010 2	001000 8	001001 9	001100 12	001101 13	011000 24	011001 25	011100 28	011101 29
011 3	001010 10	001011 11	001110 14	001111 15	011010 26	011011 27	011110 30	011111 31
100 4	100000 32	100001 33	100100 36	100101 37	110000 48	110001 49	110100 52	110101 53
101 5	100010 34	100011 35	100110 38	100111 39	110010 50	110011 51	110110 54	110111 55
110 6	101000 40	101001 41	101100 44	101101 45	111000 56	111001 57	111100 60	111101 61
111 7	101010 42	101011 43	101110 46	101111 47	111010 58	111011 59	111110 62	111111 63

X

Y

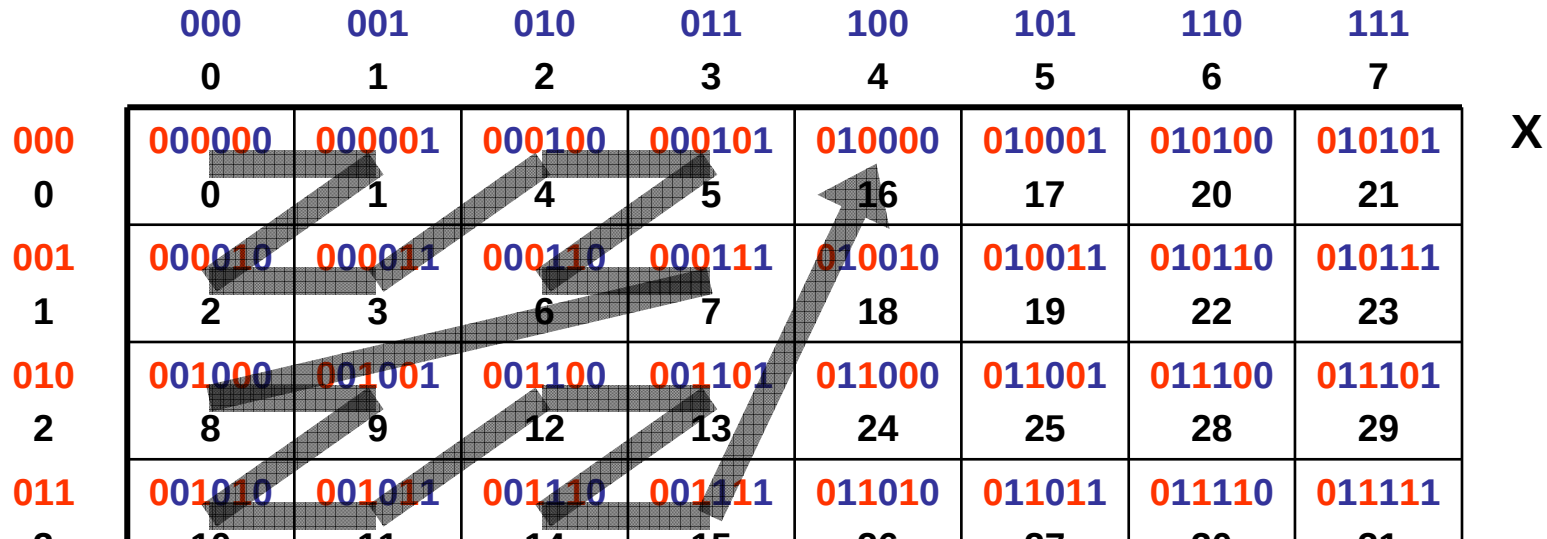
What is Z-Ordering?



Y	000	001	010	011	100	101	110	111
	0	1	2	3	4	5	6	7
000	000000	000001	000100	000101	010000	010001	010100	010101
0	0	1	4	5	16	17	20	21
001	000010	000011	000110	000111	010010	010011	010110	010111
1	2	3	6	7	18	19	22	23
010	001000	001001	001100	001101	011000	011001	011100	011101
2	8	9	12	13	24	25	28	29
011	001010	001011	001110	001111	011010	011011	011110	011111
3	10	11	14	15	26	27	30	31
100	100000	100001	100100	100101	110000	110001	110100	110101
4	32	33	36	37	48	49	52	53
101	100010	100011	100110	100111	110010	110011	110110	110111
5	34	35	38	39	50	51	54	55
110	101000	101001	101100	101101	111000	111001	111100	111101
6	40	41	44	45	56	57	60	61
111	101010	101011	101110	101111	111010	111011	111110	111111
7	42	43	46	47	58	59	62	63

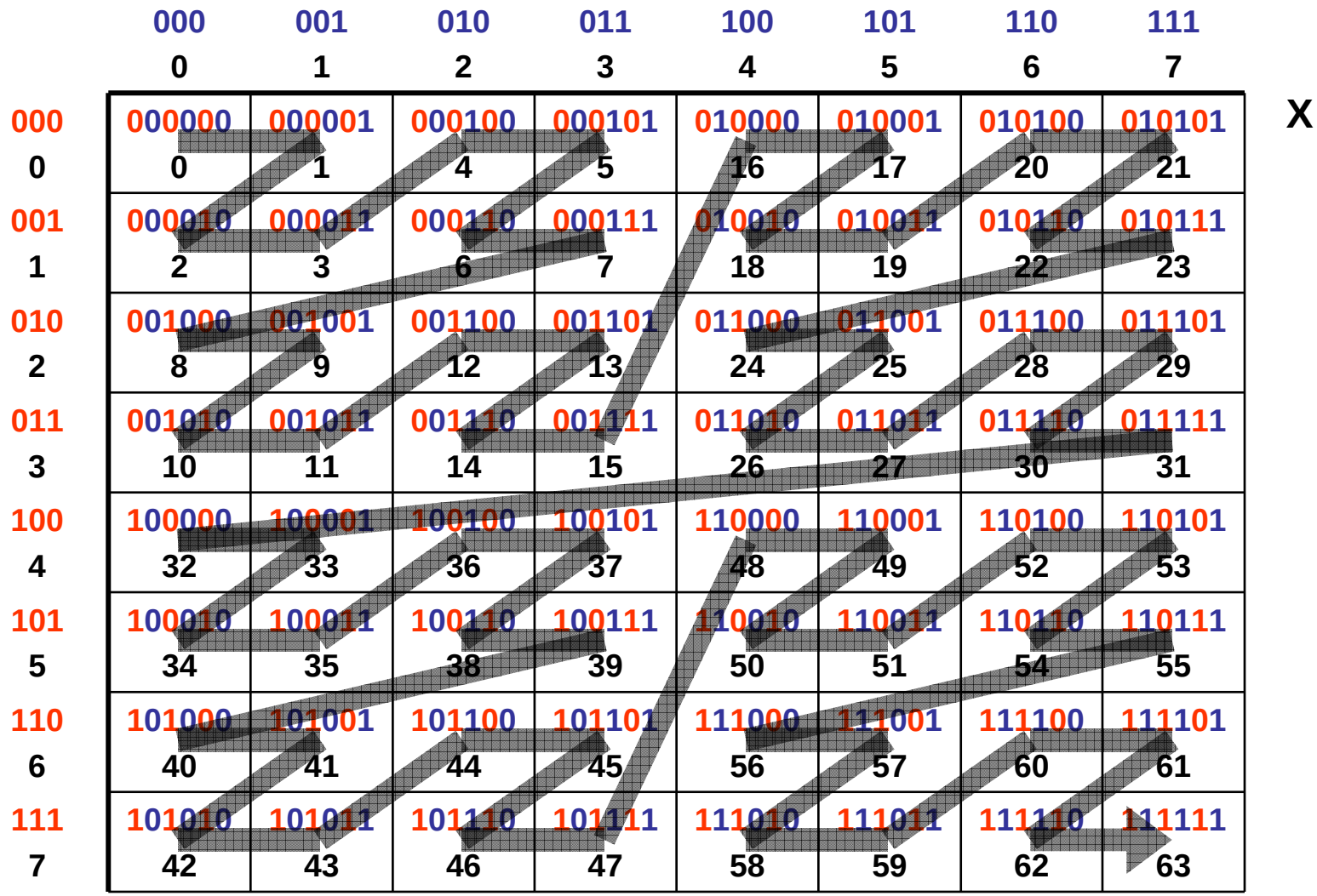
X

What is Z-Ordering?



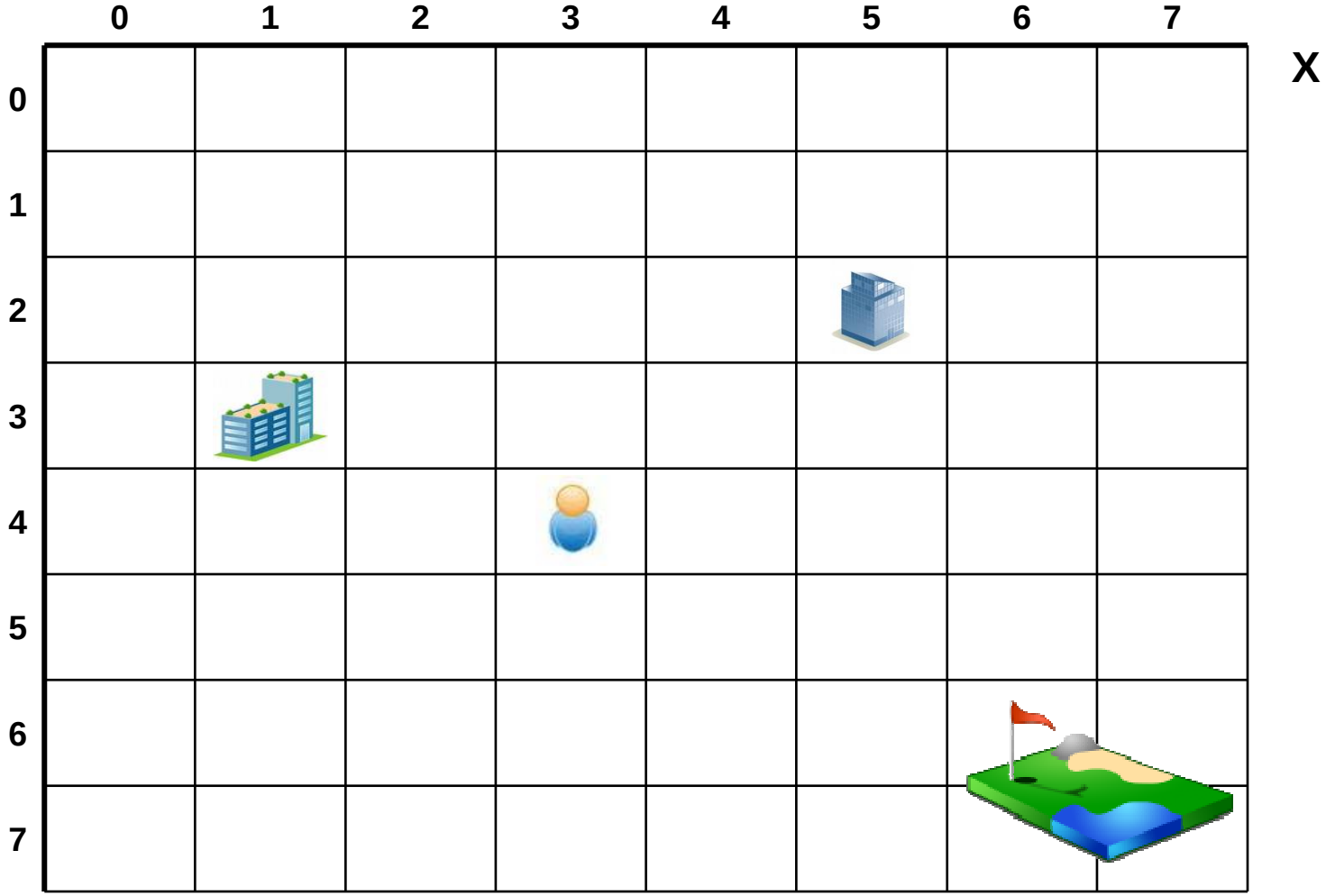
Y	X: 000 (0)	001 (1)	010 (2)	011 (3)	100 (4)	101 (5)	110 (6)	111 (7)
000 (0)	000000 0	000001 1	000100 4	000101 5	010000 16	010001 17	010100 20	010101 21
001 (1)	000010 2	000011 3	000110 6	000111 7	010010 18	010011 19	010110 22	010111 23
010 (2)	001000 8	001001 9	001100 12	001101 13	011000 24	011001 25	011100 28	011101 29
011 (3)	001010 10	001011 11	001110 14	001111 15	011010 26	011011 27	011110 30	011111 31
100 (4)	100000 32	100001 33	100100 36	100101 37	110000 48	110001 49	110100 52	110101 53
101 (5)	100010 34	100011 35	100110 38	100111 39	110010 50	110011 51	110110 54	110111 55
110 (6)	101000 40	101001 41	101100 44	101101 45	111000 56	111001 57	111100 60	111101 61
111 (7)	101010 42	101011 43	101110 46	101111 47	111010 58	111011 59	111110 62	111111 63

What is Z-Ordering?

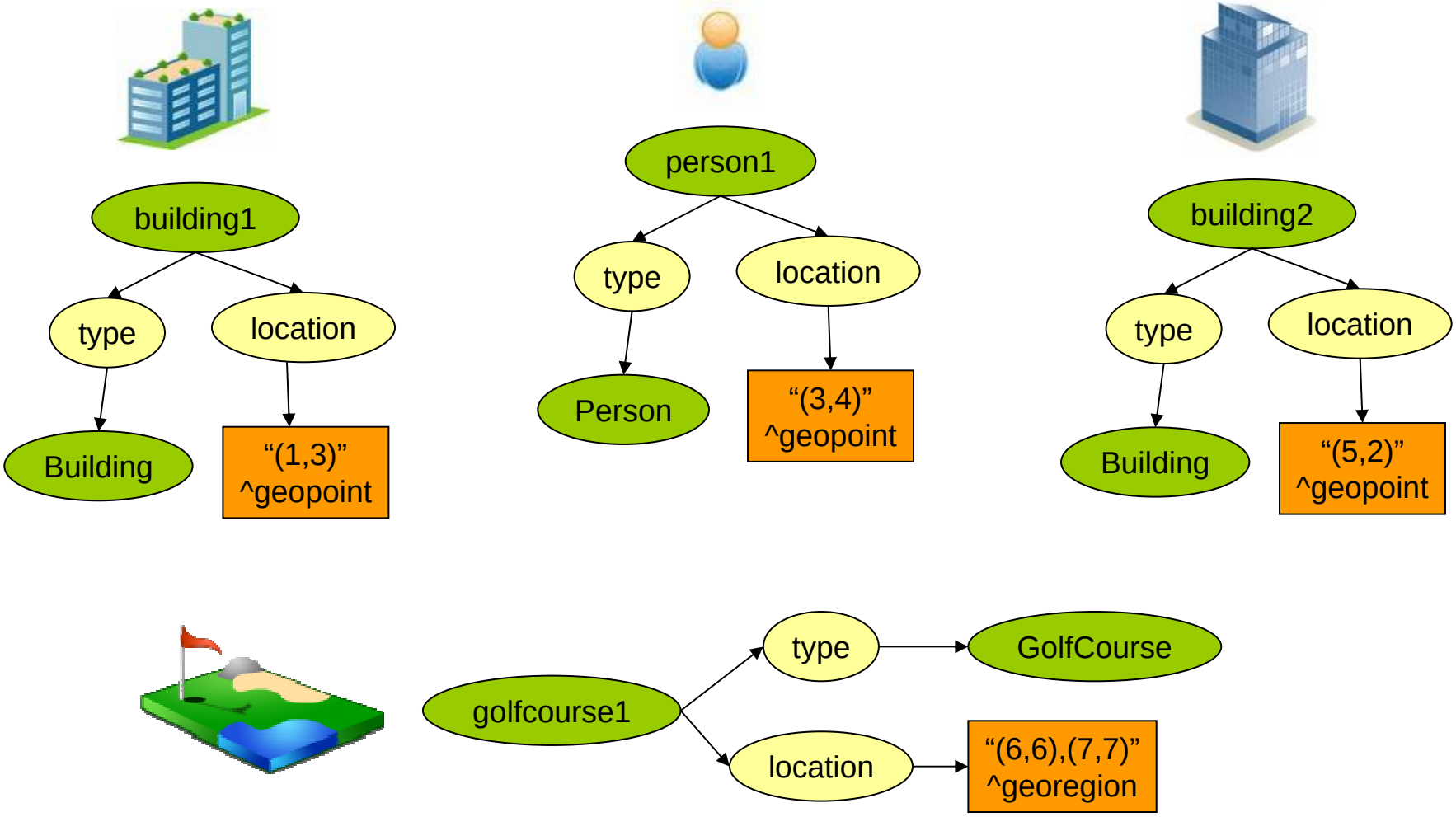


Geospatial Data in RDF

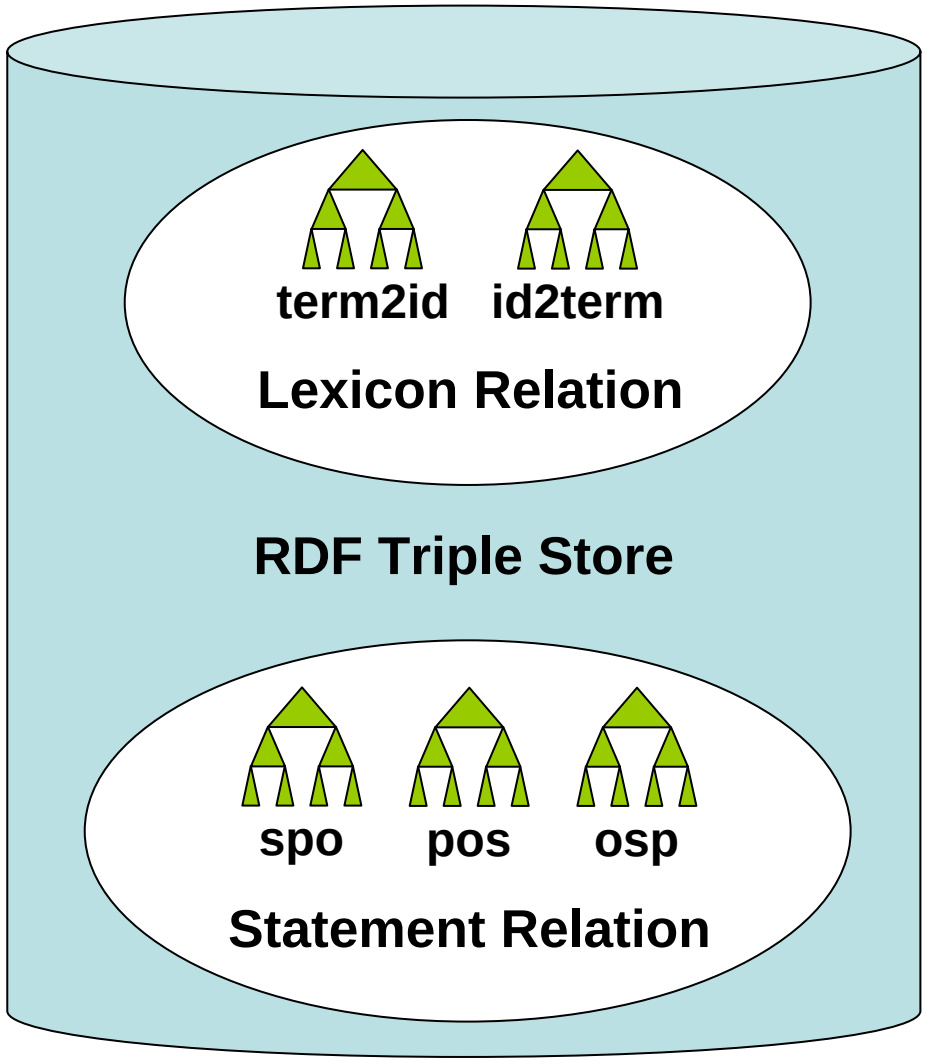
Four geospatial object instances:



Geospatial Data in RDF



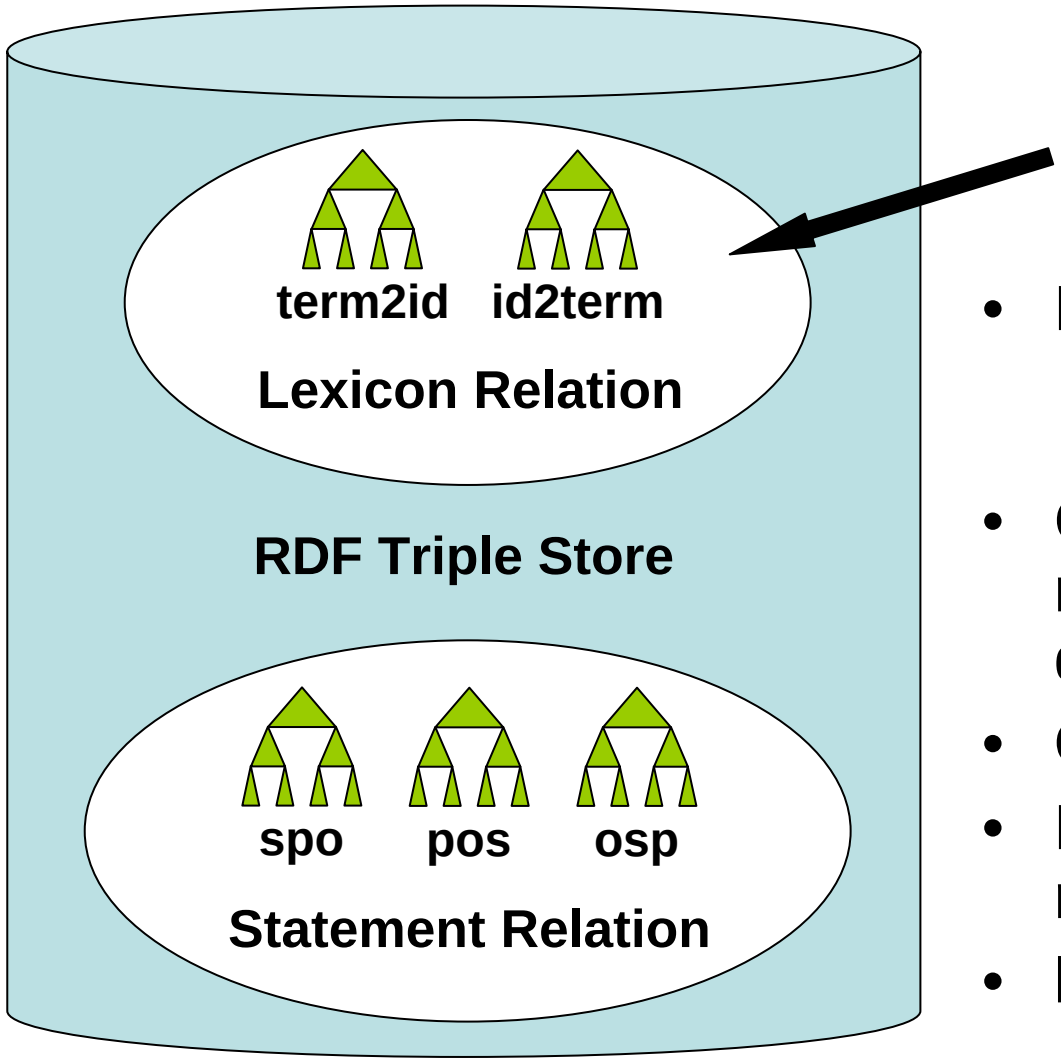
bigdata® Indexing Background



- bigdata® is all about sharded B+Trees:

- bigdata® RDF database modeled as two relations
- Lexicon relation maps RDF terms to unique long termIDs (forward and reverse)
- Statement relation provides perfect access paths for triples (3 indices) or quads (6 indices)

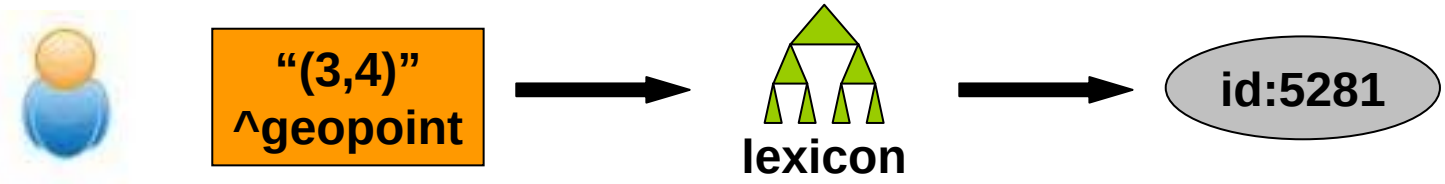
Geospatial Indexing



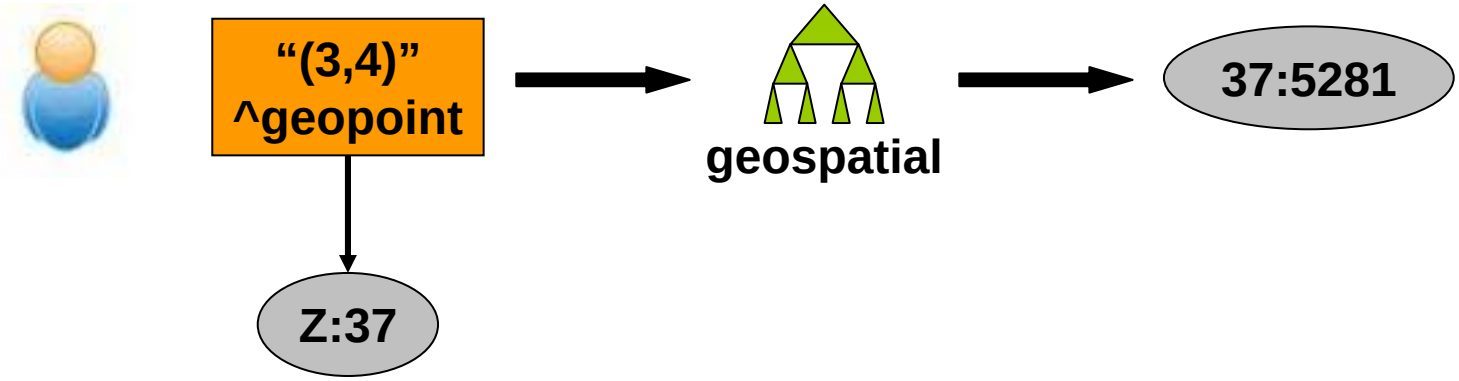
- New geospatial index
  - Actually one index per geospatial resolution
- Geospatial RDF literals marked with geospatial datatypes
- One Z-value for a point
- Many Z-values for a region
- key = [Z-value]:[termID]

Geospatial Indexing: Point

1. RDF term indexed by lexicon, unique long termID generated.

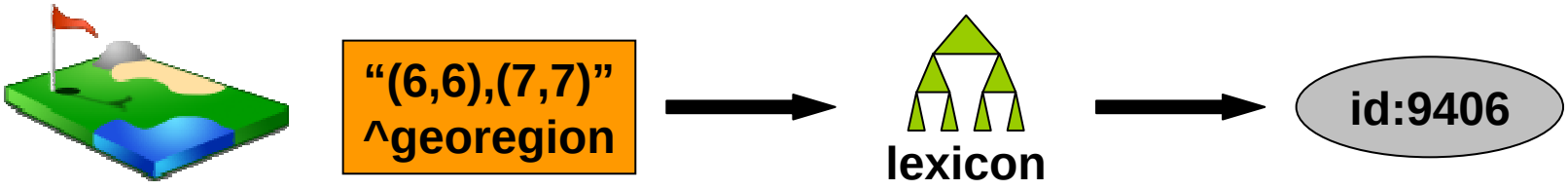


2. Z-value for point computed and indexed geospatially with termID.

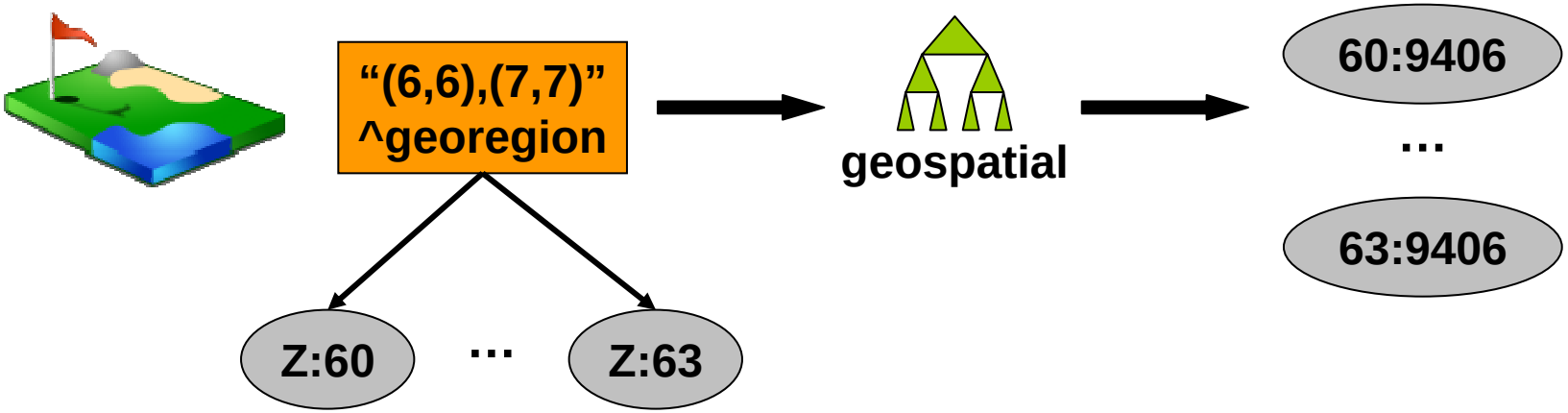


Geospatial Indexing: Region

1. Region handled similar to point, first a unique termID is generated.



2. Z-values for the region computed, each one indexed geospatially.



Geospatial Indexing

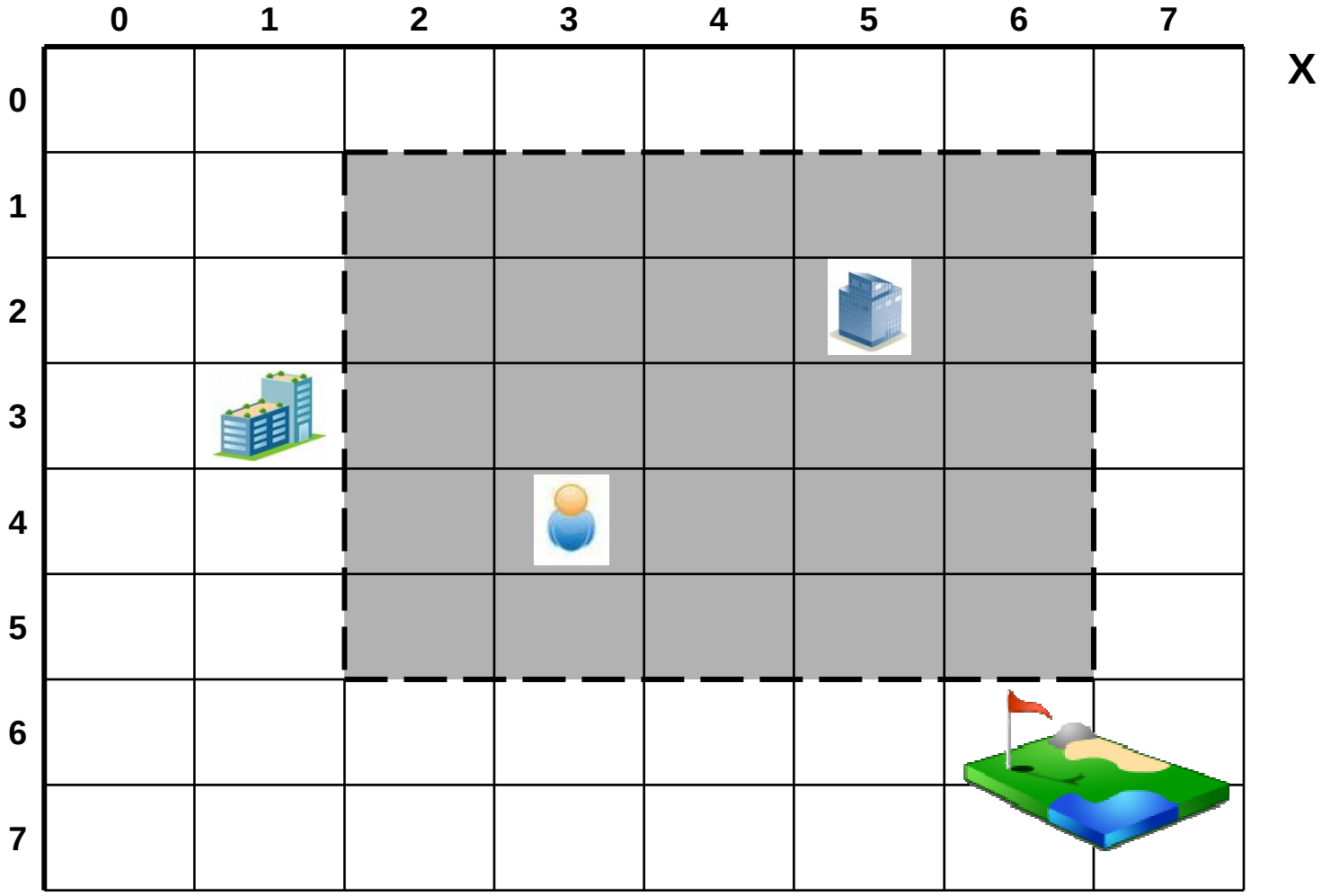
- Since Z-order provides excellent locality, and geospatial index key =

  [Z-value]:[termID]

  …geospatial data close in physical space also close in geospatial index space.
- Very nice property for index sharding!
- Better space ordering curves exist, but are patented
- R-Trees also used, but parallel descent problems make Z-ordered B-Trees better for sharding

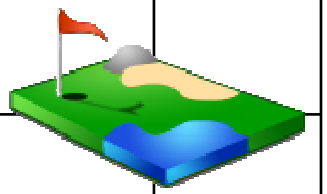
Basic Geospatial Query

"Give me everything within a selected region"



Basic Geospatial Query

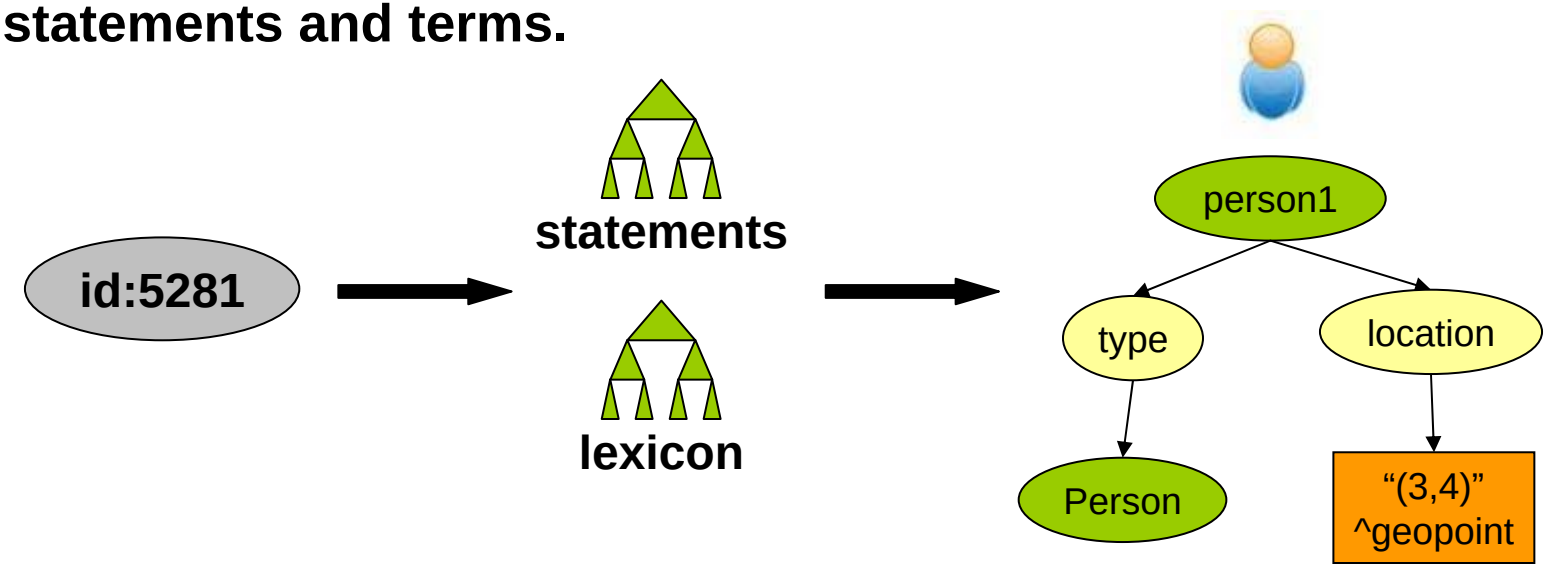
1. Compute Z-values for the selected region (linear algorithm).

Y \ X	0	1	2	3	4	5	6	7
0								
1			000110 6	000111 7	010010 18	010011 19	010110 22	
2			001100 12	001101 13	011000 24	0 1 25	011100 28	
3			001110 14	001111 15	011010 26	011011 27	011110 30	
4			100100 36	1 1 37	110000 48	110001 49	110100 52	
5			100110 38	100111 39	110010 50	110011 51	110110 54	
6								
7								

Basic Geospatial Query

2. Use geospatial index to find termIDs from Z-values.

3. Use statement and lexicon indices to find and resolve RDF statements and terms.



Geospatial Joins in SPARQL

- “Find all buildings within the selected region”

```
select ?x
where
{
  ?x type Building .
  ?x location ?loc .
  filter( geo_intersect(?loc, “(2,1),(6,5)”^georegion )) .
}
```

- Very similar to basic geospatial query
- Compute Z-values for the selected region
- Use geospatial index to find termIDs from Z-values
- Use termIDs to constrain normal SPARQL query processing

?x = building2